arXiv:1304.5804v2 [quant-ph] 25 Apr 2013

Tight Bounds on the Synthesis of 3-bit Reversible Circuits: NFT Library

Ahmed Younes*
Department of Mathematics and Computer Science
Faculty of Science, Alexandria University
Alexandria, Egypt

April 26, 2013

Abstract

The reversible circuit synthesis problem can be reduced to permutation group. This allows Schreier-Sims Algorithm for the strong generating set-finding problem to be used to find tight bounds on the synthesis of 3-bit reversible circuits using the NFT library. The tight bounds include the maximum and minimum length of 3-bit reversible circuits, the maximum and minimum cost of 3-bit reversible circuits. The analysis shows better results than that found in the literature for the lower bound of the cost. The analysis also shows that there are 1960 universal reversible sub-libraries from the main NFT library.
Keywords: Reversible circuit; Quantum Cost; Circuit Optimization; Group Theory.

1 Introduction

Reversible logic [2, 11] is one of the hot areas of research. It has many applications in quantum computation [14, 23], low-power CMOS [9, 5] and many more. Synthesis of reversible circuits cannot be done using conventional ways [29]. Synthesis and optimization of Boolean systems on non-standard computers that promise to do computation more powerfully [26] than classical computers, such as quantum computers, is an essential aim in the exploration of the benefits that may be gain from such systems.

A lot of work has been done trying to find an efficient reversible circuit for an arbitrary reversible function. In one of the research directions, it was shown that the process of synthesizing linear reversible circuits can be reduced to a row reduction problem of $n \times n$ non-singular matrix [24]. Standard row reduction methods such as Gaussian elimination and LU-decomposition have been proposed [3]. In another research direction, search algorithms and template matching tools using reversible gates libraries have been used [10, 17, 21, 22]. These will work efficiently for small circuits. A method is given in [15], where a very useful set of transformations for Boolean quantum circuits is shown. In this method, extra auxiliary bits are used in the construction that will increase the hardware cost. In [31], it was shown that there is a direct correspondence between reversible Boolean operations and certain forms of classical logic known as Reed-Muller expansions. This arises the possibility of handling the problem of synthesis and optimization of reversible Boolean logic within the field of Reed-Muller logic. A lot of work has been done trying to find an efficient reversible circuit for an arbitrary multi-output Boolean functions by using templates [18, 19] and data-structure-based optimization [25]. A method to generate an optimal 4-bit reversible circuits has been proposed [13]. Benchmarks for reversible circuits have been established [20].

Recently, the study of reversible logic synthesis problem using group theory is gaining more attention. Investigation on the universality of the basic building blocks of reversible circuit has been done [28, 6]. A relation between Young subgroups and the reversible logic synthesis problem has been proposed [7]. A comparison between the decomposition of reversible circuit and quantum circuit using group theory has been shown [8]. A GAP-based algorithms to synthesize reversible circuits for various types of gate with various gate costs has been proposed [30].

The aim of the paper is to answer the following questions for the synthesis of 3-bit reversible circuits using NFT library:

*ayounes2@yahoo.com or ayounes@alexu.edu.eg

1. What are the maximum and minimum lengths of reversible circuits for a reversible function? What are the costs of these circuits?

2. What are the maximum and minimum costs of reversible circuits for a reversible function ? What are the length of these circuits?

3. What are the upper-bound and lower-bound on lengths of reversible circuits for all 3-bit reversible functions using NFT library? What are the costs of these bounds?

4. What are the upper-bound and lower-bound on costs of reversible circuits for all 3-bit reversible functions using NFT library? What are the lengths of these bounds?

5. Is there any sub-library of NFT that can act as a universal reversible gate library? If so, what is the best universal reversible gate library that gives the best length, worst length, best cost and worst cost? How many sub-libraries that can act as a universal reversible gate library?

The paper is organized as follows: Section 2 gives a short background on the synthesis of reversible circuit problem and shows the reduction the problem to permutation group. Section 3 shows Schreier-Sims Algorithm for the strong generating set-finding problem that is used to calculate the bounds on the synthesis of 3-bit reversible circuits problem using NFT library. Section 4 shows the results of the experiments. The paper ends up with a summary and conclusion in Section 5.

2 Background

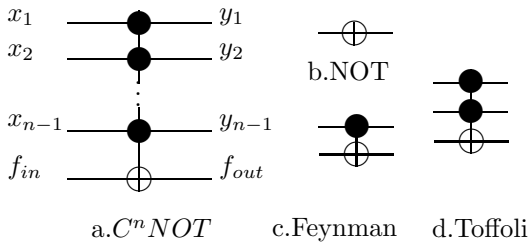


Figure 1: C^nNOT gates. The black circle • indicates the control bits, and the symbol $\oplus$ indicates the target bit. (a)C^nNOT gate with $n-1$ control bits. (b) C^1NOT gate with no control bits. (c)C^2NOT gate with one control bit. (d)C^3NOT gate with two control bits.

2.1 Reversible Circuits

To build a reversible circuit with n variables, an $n \times n$ reversible circuit is used. C^nNOT gate is the main primitive gate that is used to build the circuit since it is shown to be universal for reversible computation [29]. C^nNOT gate is defined as follows:

Definition 2.1 *(C^nNOT gate)*
C^nNOT is a reversible gate denoted as,

$$C^nNOT(x_1, x_2, \ldots, x_{n-1}; f), \tag{1}$$

with n inputs: x_1, x_2,..., x_{n-1} (known as control bits) and f_{in} (known as target bit), and n outputs: y_1, y_2,..., y_{n-1} and f_{out}. The operation of the C^nNOT gate is defined as follows,

$$\begin{aligned} y_i &= x_i, \text{for } 1 \leq i \leq n-1, \\ f_{out} &= f_{in} \oplus x_1 x_2 \ldots x_{n-1}, \end{aligned} \tag{2}$$

i.e. the target bit will be flipped if and only if all the control bits are set to 1. Some special cases of the general C^nNOT gate have their own names, C^1NOT gate with no control bits is called NOT gate as shown in Fig. 1-b, where the bit will be flipped unconditionally. C^2NOT gate with one control bit is called *Feynman* gate as shown in Fig. 1-c. C^3NOT gate with two control bits is called *Toffoli* gate as shown in Fig. 1-d.

For the sake of readability C^1NOT, C^2NOT and C^3NOT will be written for short as N, F and T respectively where the control and/or target bits will be shown in the subscript of the gate, for example, a 3-bit reversible circuit synthesized using N,F and T gates can contain 12 possible gates, as shown in Fig.2, that perform as follows:

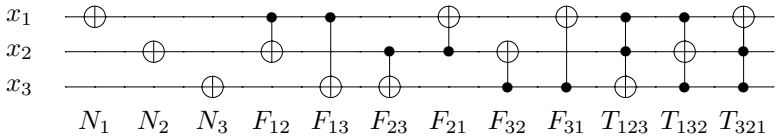


Figure 2: The 12 possible CNOT gates for a 3 bit reversible circuit.

$$
\begin{aligned}
&N_1 : (x_1, x_2, x_3) \rightarrow (x_1 \oplus 1, x_2, x_3)\\
&N_2 : (x_1, x_2, x_3) \rightarrow (x_1, x_2 \oplus 1, x_3)\\
&N_3 : (x_1, x_2, x_3) \rightarrow (x_1, x_2, x_3 \oplus 1)\\
&F_{12} : (x_1, x_2, x_3) \rightarrow (x_1, x_2 \oplus x_1, x_3)\\
&F_{13} : (x_1, x_2, x_3) \rightarrow (x_1, x_2, x_3 \oplus x_1)\\
&F_{23} : (x_1, x_2, x_3) \rightarrow (x_1, x_2, x_3 \oplus x_2)\\
&F_{21} : (x_1, x_2, x_3) \rightarrow (x_1 \oplus x_2, x_2, x_3)\\
&F_{32} : (x_1, x_2, x_3) \rightarrow (x_1, x_2 \oplus x_3, x_3)\\
&F_{31} : (x_1, x_2, x_3) \rightarrow (x_1 \oplus x_3, x_2, x_3)\\
&T_{123} : (x_1, x_2, x_3) \rightarrow (x_1, x_2, x_3 \oplus x_1x_2)\\
&T_{132} : (x_1, x_2, x_3) \rightarrow (x_1, x_2 \oplus x_1x_3, x_3)\\
&T_{321} : (x_1, x_2, x_3) \rightarrow (x_1 \oplus x_2x_3, x_2, x_3)
\end{aligned}
\tag{3}
$$

A reversible circuit is a cascade of one or more C^nNOT gates, for example, a 3-bit reversible circuit shown in Fig.4-a can be written for short as $(T_{123}F_{23}N_2T_{132}F_{13}F_{31})$, where the truth table of this circuit is shown in Fig.4-b. Another way to write this truth table is $(0,1,2,3,4,5,6,7) \rightarrow (2,6,5,4,7,1,0,3)$, or simply $(2,6,5,4,7,1,0,3)$ which is called the specification of the circuit.

2.2 Quantum Cost

Quantum cost is a term used to refer to the technological cost of building C^nNOT gates. The quantum cost of a reversible circuit is subject to optimization as well as the number of C^nNOT gates used in the circuit. The quantum cost of a C^nNOT gate is based primarily on the number of bits involved in the gate, i.e. the number of elementary operations required to build the C^nNOT gate [1].

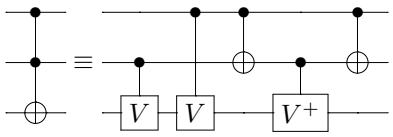


Figure 3: Decomposition of T gate as 5 elementary 2-qubit gates.

When implementing a reversible gate circuit on a quantum computer, four elementary quantum gates are used [1]: N gate, F gate, Controlled-V and Controlled-V^+ gates, where $VV = V^+V^+ = N$, $VV^+ = V^+V = I$, where I is the identity. The cost of a 2-qubit gate is much larger than that of a 1-qubit gate. In this paper, the cost of N gate is ignored as in [30] to be able to compare results, and assume that the cost of any 2-qubit gate is equal to 1. The cost of a reversible circuit is measured by the number of 2-qubit gates used in its implementation as a quantum circuit, so, the quantum cost of T gate is 5 as shown in Fig.3.

2.3 Basic Notions

In this section, the basic notions for reversible circuit, permutation group, NFT library, the relationship between reversible logic circuits and permutation group theory will be reviewed.

Definition 2.2 *Let $X = \{0, 1\}$. A Boolean function f with n input variables, $x_1, \ldots, x_n$, and n output variables, $y_1, \ldots, y_n$, is a function $f : X^n \to X^n$, where $(x_1, \ldots, x_n) \in X^n$ is called the input vector and $(y_1, \ldots, y_n) \in X^n$ is called the output vector.*

Definition 2.3 *An n-input n-output Boolean function is reversible ($n \times n$ function) if it maps each input vector to a unique output vector, i.e. a one-to-one, onto function (bijection).*

There are $2^n!$ reversible $n \times n$ Boolean functions. For $n = 3$, there are 40,320 3-input 3-output reversible functions.

Definition 2.4 *An n-input n-output reversible gate (or circuit) is a gate that realizes an $n \times n$ reversible function.*

Definition 2.5 *A set of reversible gates that can be used to build a reversible circuit is called a gate library L.*

Definition 2.6 *A universal reversible gate library L_n is a set of reversible gates that can be used to build any reversible circuit with n-input n-output.*

Definition 2.7 *Consider a finite set $A = \{1, 2, ..., N\}$ and a bijection $\sigma : A \to A$, then σ can be written as,*

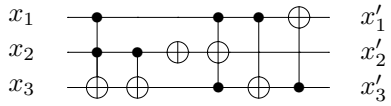
$$\sigma = \begin{pmatrix} 1 & 2 & 3 & ... & N \\ \sigma(1) & \sigma(2) & \sigma(3) & ... & \sigma(N) \end{pmatrix}, \tag{4}$$

i.e. σ is a permutation of A. Let A be an ordered set, then the top row can be eliminated and σ can be written as,

$$(\sigma(1), \sigma(2), \sigma(3), ..., \sigma(N)). \tag{5}$$

Any reversible circuit with n-input n-output can be considered as a permutation σ and Eqn.5 is called the specification of this reversible circuit such that $N = 2^n$.

a.



b.

x_1	x_2	x_3	x'_1	x'_2	x'_3
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	0	0	1
1	1	0	0	0	0
1	1	1	0	1	1

Figure 4: (a.) Circuit implementation of a reversible function. (b.) Reversible truth table.

The set of all permutations on A forms a symmetric group on A under composition of mappings [16], denoted by S_N [4]. A permutation group G is a subgroup [16] of the symmetric group S_N. A universal reversible gate library L_n is called the generators of the group. Another important notation of a permutation is the product of disjoint cycles [4]. For example, $\begin{pmatrix} 1,2,3,4,5,6,7,8 \\ 3,2,5,4,6,1,8,7 \end{pmatrix}$ will be written as $(1, 3, 5, 6)(7, 8)$. The identity mapping "()" is called the unit element in a permutation group. A product $p * q$ of two permutations p and q means applying mapping p then q, which is equivalent to cascading p and q.

The one-to-one correspondence between a $n \times n$ reversible circuit and a permutation on $A = \{1, 2, ..., N\}$ is established as done in [30]. In the permutation group references, A begins from one, instead of zero. Therefore, we have the following relation [30]: $< X_N, X_{N-1}, \ldots, X_1 >_2 = index(X_N, \ldots, X_1) - 1$. Using the integer coding, a permutation is considered as a bijective function $f : \{1, 2, \ldots, N\} \to \{1, 2, \ldots, N\}$.

Cascading two generators is equivalent to multiplying two permutations. In what follows, a $n \times n$ reversible gate is not distinguished from a permutation in S_N. The set of 12 gates shown in Fig. 2 and performs as shown in Eqn.3 can be re-written as product of disjoint cycles as follows,

$$\begin{array}{l} N_1 : (1,5)(2,6)(3,7)(4,8) \\ N_2 : (1,3)(2,4)(5,7)(6,8) \\ N_3 : (1,2)(3,4)(5,6)(7,8) \\ F_{12} : (5,7)(6,8) \\ F_{13} : (5,6)(7,8) \\ F_{23} : (3,4)(7,8) \\ F_{21} : (3,7)(4,8) \\ F_{32} : (2,4)(6,8) \\ F_{31} : (2,6)(4,8) \\ T_{123} : (7,8) \\ T_{132} : (6,8) \\ T_{321} : (4,8) \end{array} \tag{6}$$

3 Schreier-Sims Algorithm

Let A be a finite set. The Symmetric group, $Sym(A)$, is the group of all bijections from A to itself. A permutation group G is a subgroup of $Sym(A)$. Let $G \leq Sym(A)$. For $\alpha \in A$, let G_α denotes the stabilizer of α in G, i.e., $G_\alpha = \{g \in G | \alpha^g = \alpha$. Let $B = (\alpha_1, \alpha_2, \ldots, \alpha_k)$ with $\alpha_i \in G$, then B is a base for G if $G_{(\alpha_1,\alpha_2,\ldots,\alpha_k)} = 1$, and the chain of subgroups,

$$G = G^{(1)} \geq G^{(2)} \geq \ldots \geq G^{(k+1)} = \{1\}, \tag{7}$$

defined by $G^{(i+1)} = G^{(i)}_{\alpha_i}$ for $1 \leq i \leq k$ is called the stablizer chain for B. The orbit of α under G, denoted α^G is the set $\alpha^G = \{\alpha^g | g \in G\}$. A strong generating set for G relative to B, denoted S, is a set $S \subseteq G$ and for every i with $1 \leq i \leq k+1$, $G^{(i)} = \left\langle S \cap G^{(i)} \right\rangle$ holds.

This allows the problem of synthesizing a minimal number of gates (generators) to be reduced to a strong generating set-finding problem, that is, given a group G acting faithfully on a finite set A of size N. G is specified by means of a generating set S where each element of S is expressed as a permutation on A.

Schreier-Sims Algorithm is a poly-time algorithm [27] known in computational Group Theory that solves the strong generating set-finding problem and also go further to solve membership testing problem. Schreier-Sims Algorithm keeps computing the generating sets using cosets, by trimming down the size of the generating set at every step using a depth first search approach to keep the size of the generating set from growing too large. An implementation of Schreier-Sims Algorithm on GAP [12] has been used to find the minimal number of generators that generates a specification of a reversible circuit given that the generators of the group can represent that specification .

Given the universal gate library NFT with 12 generators as shown in Eqn.6 and the 40320 specifications for all 3-input 3-output reversible circuits. The aim of the experiment is to answer the questions shown in the aim of the paper. To answer these questions, all sub-libraries of NFT library has been generated, that is 4095 sub-libraries after excluding the identity mapping. Use every sub-library to try to synthesize a reversible circuit for the 40320 specifications, if possible, using Schreier-Sims Algorithm. The term "if possible" here means that if a specification does not belong to the group generated by a sub-library, then it is impossible for this specification to be represented as a reversible circuit using this sub-library. The process of synthesizing all possible 3-bit reversible circuits is shown in Algorithim 1.

Algorithm 1 Generation of All 3 bits reversible circuits

```
1: procedure ()
2:     for i ← 1, NumOfSubLibraries do
3:         for j ← 1, NumOfCircuits do
4:             if Specs[j] ∈ G[i] then
5:                 Circuit[i][j] = SchreierSims(G[i], Specs[j])
6:             else
7:                 Circuit[i][j] = []
8:             end if
9:         end for
10:    end for
11: end procedure
```

4 Expermimental Results

Among the $4,095 \times 40,319 = 165,106,305$ trials to synthesize a circuit for a specification using a sub-library, 80,925,627 circuits have been synthesized. The trivial specification "()" and the trivial identity mapping have been ignored. The results have been organized to answer the specification related questions (Question 1 to Question 4) in Section 4.1, and the question related to the sub-libraries (Question 5) has been answered in Section 4.2.

4.1 Specification Oriented Results

In this section, the results are organized to show a summary for the specification oriented results in Table 1. The results related to the bounds on circuits length are shown in Tables 2, 3, 4 and 5. Then the results related to the bounds on circuits cost are shown in Tables 6, 7, 8 and 9.

#lib	#specs
1960	29670
1984	216
2016	1746
2044	36
2080	1170
2085	1
2086	559
2116	540
2120	96
2122	2974
2128	162
2132	192
2144	78
2152	36
2176	24
2191	27
2196	96
2198	12
2217	1293
2218	108
2220	72
2246	12
2249	36

#lib	#specs
2263	108
2264	24
2266	80
2268	12
2274	12
2284	48
2287	135
2311	27
2320	210
2324	24
2335	9
2348	12
2353	24
2354	12
2393	45
2412	72
2417	27
2428	24
2432	11
2450	6
2455	27
2456	3
2465	3

#lib	#specs
2468	12
2496	6
2525	24
2528	27
2540	6
2560	27
2605	12
2624	13
2625	8
2636	6
2676	12
2688	9
2689	3
2705	3
2732	6
2816	7
2880	6
2944	6
2961	3
3072	3
3200	3
3264	6

Table 1: There are *#specs* specifications that can be synthesized by *#lib* sub-libraries.

Table 1 shows the number of sub-libraries that can be used to synthesize a circuit for a specification, for example, there are 1960 sub-libraries that can be used to synthesize a circuit for 29670 specifications. There are 6 popular specifications that can be synthesized by 3264 different sub-libraries. These 6 specifications are the specifications of $F_{12}, F_{13}, F_{23}, F_{21}, F_{32}$ and F_{31}. Fig.5 shows the worst circuit

representation for F_{31} with 10 gates and cost = 24. This shows that any F gates can be removed from a library and be replaced by another set of generators although there might be an increase in the cost of the circuit.

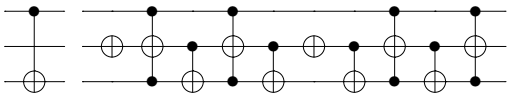


Figure 5: The worst circuit representation for F_{31}.

Max Len	#specs
7	3
8	7
9	16
10	29
11	63
12	154
13	1455
14	14966
15	19544
16	3815
17	243
18	24

Table 2: There are *#specs* that have circuits with maximum length = $Maxlen$. The average maximum length = 14.639.

Table 2 shows the maximum length for the 40319 specifications, after excluding the identity. There are 3 specifications with maximum length = 7 gates and 24 specifications with maximum length = 18 gates. The average maximum length is 14.639. Table 3 shows the same results as Table 2 with more details on the cost of the maximum length, for example, there are 3 specifications with maximum length = 7 gates, the cost of two of them is 9, while the third specification has cost =11.

Table 4 shows the minimum length for the 40319 specifications, after excluding the identity. There are 12 specifications with minimum length = 1 gate, these are the generators shown in Eqn.6, while there are 577 specifications with minimum length = 8 gates. These results are identical with that shown in [30]. This comes from the fact that all minimum length circuits come from using the gate library that contains all the 12 generators shown in Eqn.6. This is not true when finding a library of gates that synthesize minimum cost circuits as shown later. Table 5 shows the same results as Table 4 with more details on the cost of the minimum length circuit.

Table 6 shows the maximum cost for the 40319 specifications, after excluding the identity. There is 1 specification with maximum cost = 18 and 2 specifications with maximum cost = 54. The average maximum cost is 40.759. Table 7 shows the same results as Table 2 with more details on the length of the maximum cost, for example, the length of the only specification with maximum cost = 18 is 6 gates and the length of the two specifications with maximum cost = 54 are 15 gates.

Table 8 shows the minimum cost for the 40319 specifications, after excluding the identity. There are 7 specifications with minimum cost = 0, these are the cost of specifications synthesized by circuits contain the gates N_1, N_2 and N_3 only. There are 7 such circuits starting from a circuit with only one N gate to a circuit consists of the three gates: $(N_1N_2N_3)$. There are 2049 specifications with minimum cost = 17. The average minimum cost is 11.769. This is better that that shown in [30], this enhancement comes from considering all sub-libraries. Table 9 shows the same results as Table 8 with more details on the length of the circuits with minimum cost.

Comparing the above results related to the bounds on circuits length with the results related to the bound on circuits cost shows that the cost and the length of a circuit using a library is not correlated if all types of generators are considered, not only the gates used to calculate the cost, e.g. T gate is counted as 1 in circuit length while it is counted 5 in circuit cost. For example, when searching for the best length circuit for the specification $((1,7)(2,5)(3,6,8,4))$, the best length circuit is $(F_{12}T_{321}T_{123}N_1F_{13}F_{32}T_{123})$

Max Len	cost	#specs
7	9	2
7	11	1
8	14	3
8	20	4
9	11	1
9	13	2
9	17	3
9	19	6
9	21	1
9	24	3
10	16	3
10	18	1
10	22	9
10	24	9
10	26	7
11	11	2
11	17	6
11	19	1
11	21	12
11	22	2
11	23	14
11	24	2
11	26	6
11	27	18
12	14	6
12	16	8
12	17	2
12	18	21
12	21	14
12	22	27
12	23	1
12	24	7
12	26	17
12	27	8
12	28	4
12	29	8
12	30	2
12	31	6
12	32	10
12	33	4
12	34	6
12	38	3
13	15	5
13	16	2
13	17	49
13	18	52

Max Len	cost	#specs
13	19	64
13	20	37
13	21	40
13	22	187
13	23	213
13	24	72
13	25	13
13	26	59
13	27	148
13	28	124
13	29	29
13	30	43
13	31	74
13	32	82
13	33	11
13	34	7
13	35	19
13	36	50
13	37	28
13	38	19
13	39	20
13	40	2
13	41	4
13	43	2
14	16	7
14	18	110
14	19	132
14	20	157
14	21	109
14	22	662
14	23	1735
14	24	1275
14	25	254
14	26	404
14	27	2081
14	28	3115
14	29	976
14	30	129
14	31	493
14	32	1389
14	33	702
14	34	46
14	35	86
14	36	137
14	37	263
14	38	126

Max Len	cost	#specs
14	39	122
14	40	107
14	41	109
14	42	62
14	43	46
14	44	52
14	45	48
14	46	9
14	47	1
14	48	16
14	49	6
15	19	24
15	20	29
15	21	26
15	22	102
15	23	1126
15	24	1392
15	25	473
15	26	76
15	27	1364
15	28	3963
15	29	2517
15	30	376
15	31	277
15	32	1653
15	33	2718
15	34	1047
15	35	60
15	36	115
15	37	317
15	38	361
15	39	137
15	40	174
15	41	222
15	42	219
15	43	195
15	44	238
15	45	98
15	46	63
15	47	32
15	48	47
15	49	57
15	50	22
15	51	6
15	53	18
16	20	3

Max Len	cost	#specs
16	22	3
16	23	18
16	24	251
16	25	102
16	26	6
16	27	66
16	28	335
16	29	505
16	30	189
16	31	15
16	32	141
16	33	728
16	34	836
16	35	167
16	36	9
16	37	16
16	38	128
16	39	97
16	40	17
16	41	34
16	42	17
16	43	16
16	44	31
16	45	22
16	46	14
16	47	17
16	48	6
16	49	21
16	50	2
16	51	1
16	53	2
17	25	12
17	28	18
17	29	3
17	32	9
17	33	12
17	34	63
17	35	66
17	36	6
17	39	45
17	40	9
18	29	3
18	33	6
18	36	6
18	40	9

Table 3: There are *#specs* specifications with maximum length= $MaxLen$ and cost = *cost*.

Min Len	#specs	#specs [30]
1	12	12
2	102	102
3	625	625
4	2780	2780
5	8921	8921
6	17049	17049
7	10253	10253
8	577	577
Avg.	5.865	5.865

Table 4: There are *#specs* specifications with minimum length = $MinLen$.

Min Len	cost	#specs
1	0	3
1	1	6
1	5	3
2	0	3
2	1	24
2	2	24
2	5	15
2	6	30
2	9	3
2	10	3
3	0	1
3	1	18
3	2	117
3	3	51
3	5	24
3	6	162
3	7	138
3	9	18
3	10	51
3	11	39
3	14	5
3	15	1
4	2	51
4	3	282
4	4	60
4	5	8

Min Len	cost	#specs
4	6	186
4	7	772
4	8	369
4	9	39
4	10	228
4	11	430
4	12	233
4	14	45
4	15	62
4	16	13
4	19	2
5	3	75
5	4	375
5	5	24
5	6	10
5	7	673
5	8	2041
5	9	516
5	10	284
5	11	1181
5	12	1883
5	13	669
5	14	137
5	15	490
5	16	409
5	17	89

Min Len	cost	#specs
5	18	5
5	19	45
5	20	14
5	21	1
6	4	45
6	5	168
6	6	2
6	7	22
6	8	1219
6	9	2233
6	10	253
6	11	833
6	12	3070
6	13	3933
6	14	759
6	15	913
6	16	1860
6	17	1242
6	18	234
6	19	142
6	20	104
6	21	16
6	22	1
7	6	14
7	8	33
7	9	518

Min Len	cost	#specs
7	10	222
7	11	106
7	12	856
7	13	2278
7	14	1200
7	15	508
7	16	1493
7	17	1813
7	18	848
7	19	226
7	20	101
7	21	28
7	22	3
7	23	3
7	24	3
8	9	3
8	12	9
8	13	31
8	14	36
8	15	22
8	16	145
8	17	173
8	18	109
8	19	39
8	20	9
8	21	1

Table 5: There are *#specs* specifications with minimum length= $MinLen$ and cost = *cost*.

Max Cost	#specs
18	1
19	1
20	1
22	10
23	9
24	21
25	22
26	18
27	41
28	23
29	37
30	66

Max Cost	#specs
31	122
32	314
33	549
34	525
35	972
36	1816
37	3088
38	3035
39	3769
40	4111
41	3854
42	4067

Max Cost	#specs
43	5406
44	3484
45	1652
46	883
47	687
48	914
49	622
50	96
51	28
52	5
53	68
54	2

Table 6: There are *#specs* specifications with maximum cost = $MaxCost$. The average maximum cost is 40.759.

Max Cost	Len	#specs
18	6	1
19	6	1
20	6	1
22	7	1
22	10	3
22	11	3
22	12	3
23	7	2
23	8	4
23	11	2
23	12	1
24	7	4
24	8	7
24	9	4
24	10	3
24	11	2
24	12	1
25	7	18
25	8	2
25	9	2
26	7	5
26	8	2
26	9	1
26	10	10
27	7	14
27	8	2
27	9	5
27	11	18
27	12	2
28	8	8
28	9	3
28	10	3
28	11	6
28	12	3
29	7	2
29	8	1
29	9	4
29	10	7
29	11	6
29	12	17
30	8	2
30	9	3
30	10	38
30	11	7
30	12	15

Max Cost	Len	#specs
30	13	1
31	7	1
31	9	1
31	10	25
31	11	29
31	12	33
31	13	27
31	14	5
31	15	1
32	8	1
32	10	36
32	11	53
32	12	98
32	13	73
32	14	44
32	15	9
33	8	2
33	9	15
33	10	18
33	11	54
33	12	193
33	13	173
33	14	51
33	15	43
34	8	1
34	9	1
34	10	16
34	11	88
34	12	186
34	13	168
34	14	41
34	15	24
35	9	2
35	10	8
35	11	127
35	12	370
35	13	406
35	14	59
36	10	2
36	11	98
36	12	556
36	13	839
36	14	292
36	15	29
37	10	44

Max Cost	Len	#specs
37	11	207
37	12	575
37	13	1268
37	14	864
37	15	127
37	16	3
38	10	72
38	11	376
38	12	745
38	13	972
38	14	642
38	15	181
38	16	47
39	10	24
39	11	305
39	12	983
39	13	1467
39	14	830
39	15	132
39	16	28
40	10	2
40	11	83
40	12	631
40	13	1682
40	14	1437
40	15	261
40	16	15
41	11	58
41	12	378
41	13	1297
41	14	1685
41	15	411
41	16	25
42	11	107
42	12	854
42	13	1391
42	14	1305
42	15	393
42	16	17
43	11	87
43	12	1251
43	13	2241
43	14	1457
43	15	355
43	16	15

Max Cost	Len	#specs
44	14	1652
44	13	946
44	15	494
44	16	28
44	12	351
44	11	13
45	11	1
45	12	98
45	13	215
45	14	921
45	15	394
45	16	23
46	12	31
46	13	109
46	14	405
46	15	323
46	16	15
47	12	22
47	13	168
47	14	275
47	15	204
47	16	18
48	13	147
48	14	529
48	15	230
48	16	8
49	13	1
49	14	296
49	15	303
49	16	22
50	13	1
50	14	10
50	15	83
50	16	2
51	14	9
51	15	18
51	16	1
52	14	1
52	15	4
53	14	22
53	15	44
53	16	2
54	15	2

Table 7: There are *#specs* specifications with maximum cost = $MaxCost$ and length = Len.

Min Cost	#specs	#specs[30]
0	7	7
1	48	48
2	192	192
3	408	408
4	480	480
5	288	288
6	592	592
7	1962	2016
8	3887	4128
9	2916	2496
10	1299	672
11	3683	2880
12	7221	7488
13	6059	7488
14	1465	384
15	3562	1600
16	4201	5568
17	2049	3584
Avg.	11.76967	11.98313

Table 8: There are $\#specs$ specifications with minimum cost = $MinCost$.

Min Cost	Len	#specs
0	1	3
0	2	3
0	3	1
1	1	6
1	2	24
1	3	18
2	2	24
2	3	117
2	4	51
3	3	51
3	4	282
3	5	75
4	4	60
4	5	375
4	6	45
5	1	3
5	2	15
5	3	30
5	4	30
5	5	39
5	6	171
6	2	24
6	3	132
6	4	222
6	5	150
6	6	50
6	7	14
7	3	99

Min Cost	Len	#specs
7	4	615
7	5	820
7	6	362
7	7	63
7	8	3
8	4	249
8	5	1649
8	6	1489
8	7	446
8	8	48
8	9	6
9	2	3
9	3	18
9	4	45
9	5	356
9	6	1794
9	7	600
9	8	97
9	9	2
9	10	1
10	2	3
10	3	39
10	4	180
10	5	363
10	6	391
10	7	295
10	8	28
11	3	27

Min Cost	Len	#specs
11	4	261
11	5	962
11	6	1366
11	7	801
11	8	248
11	9	18
12	4	123
12	5	1090
12	6	2738
12	7	2233
12	8	869
12	9	165
12	10	3
13	5	276
13	6	2244
13	7	2158
13	8	875
13	9	381
13	10	113
13	11	12
14	3	5
14	4	41
14	5	141
14	6	360
14	7	473
14	8	247
14	9	125
14	10	58

Min Cost	Len	#specs
14	11	13
14	12	2
15	3	1
15	4	35
15	5	295
15	6	869
15	7	1111
15	8	805
15	9	383
15	10	63
16	4	8
16	5	154
16	6	873
16	7	1443
16	8	1023
16	9	591
16	10	109
17	5	26
17	6	290
17	7	548
17	8	332
17	9	347
17	10	338
17	11	155
17	12	13

Table 9: There are $\#specs$ specifications with minimum cost = $MinCost$ and length = Len.

with length = 7 and quantum cost = 18, while searching for the best cost circuit for the same specification, the best cost circuit is $(N_2N_3T_{321}N_2F_{13}T_{321}N_2N_3F_{12}T_{321}N_2N_3)$ with length = 12 and cost = 17.

4.2 Library Oriented Results

In this section, the results are organized to show a summary of library oriented results in Table 10. The results related to the bounds on circuits maximum length synthesized by the sub-libraries are shown in Tables 11, 12 and 13. Then results related to the bounds on circuits maximum cost synthesized by the sub-libraries are shown in Tables 14, 15 and 16. The results of minimum cost and minimum circuit length are trivial results, since all sub-libraries can synthesize a circuit of length 1 and cost 0, 1 or 5.

#lib	#specs
1	4
1	1342
5	5
9	35
12	1
15	11
18	167
24	71
24	119
27	143
30	3
37	23
72	383
73	7

#lib	#specs
90	15
90	31
90	47
96	63
96	127
99	95
108	575
117	719
125	1343
162	5039
168	1439
186	191
360	1151
1960	40319

Table 10: There are $\#Lib$ sub-libraries that can synthesize a circuit for $\#specs$ specifications.

Table 10 shows the ability of a sub-library to synthesize a circuit for a specification. It can be seen that there are 1960 sub-libraries that can be used to synthesize a circuit of any specification, i.e. there are 1960 universal reversible sub-libraries from the main library shown in Eqn.6. It is shown in the previous section that the main library is the best library to synthesize a minimum length circuit, while choosing a sub-library to synthesize a minimal cost circuit is not a trivial task.

Max Len	#lib
1	12
2	54
3	79
4	148
5	231
6	289
7	223
8	364
9	453

Max Len	#lib
10	498
11	498
12	433
13	378
14	255
15	120
16	51
17	6
18	3

Table 11: There are $\#lib$ sub-libraries that can synthesize circuits with maximum length = $MaxLen$.

Table 11 shows the maximum length circuits synthesized by a sub-library. There are 12 sub-libraries that can synthesize a circuit with maximum length = 1. These are the sub-libraries that each of them contain a single generator from the main library shown in Eqn.6, where there are 3 sub-libraries that synthesize a circuit with maximum length = 18. These three sub-libraries are $\{N_3, F_{32}, F_{31}, T_{123}\}$,$\{N_2, F_{23}, F_{21}, T_{132}\}$ and $\{N_1, F_{12}, F_{13}, T_{321}\}$. These sub-libraries are three examples from the 1960 universal reversible sub-libraries. Table 12 continued in Table 13 shows the same results as Table 11 with more details on the cost of the circuits synthesized by these sub-libraries.

Table 14 shows the maximum cost circuits synthesized by a sub-library. There are 7 sub-libraries that can synthesize a circuit with maximum cost = 0. These are the sub-libraries that each of them contain a combination of N generators, where there are 2 sub-libraries that synthesize a circuit with maximum cost = 54. These two sub-libraries are $\{N_1, N_3, F_{32}, T_{123}, T_{321}\}$ and $\{N_1, F_{12}, T_{123}, T_{321}\}$. These are another

Max Len	cost	#lib
1	0	3
1	1	6
1	5	3
2	0	3
2	1	18
2	2	12
2	5	3
2	6	18
3	0	1
3	1	12
3	2	21
3	3	3
3	6	24
3	7	15
3	14	2
3	15	1
4	2	40
4	3	9
4	4	33
4	6	12
4	7	17
4	8	7
4	10	12
4	12	11
4	15	4
4	16	2
4	18	1
5	2	11
5	3	43
5	4	39
5	6	4
5	7	5
5	8	18
5	9	10
5	10	34
5	11	23
5	12	19
5	13	5
5	15	12
5	16	6
5	19	1
5	20	1
6	3	11
6	4	76
6	6	7

Max Len	cost	#lib
6	7	15
6	8	26
6	9	21
6	10	36
6	11	23
6	12	38
6	13	5
6	14	11
6	15	3
6	16	3
6	18	8
6	20	2
6	21	2
6	22	2
7	5	39
7	6	10
7	7	6
7	8	14
7	9	27
7	11	12
7	12	4
7	13	22
7	14	6
7	15	17
7	16	5
7	17	13
7	18	2
7	19	27
7	20	4
7	22	4
7	25	4
7	26	3
7	27	4
8	6	32
8	7	10
8	8	5
8	10	59
8	11	2
8	12	5
8	13	25
8	14	9
8	15	28
8	16	24
8	17	15
8	18	45

Max Len	cost	#lib
8	19	42
8	20	22
8	21	1
8	22	10
8	23	10
8	24	13
8	25	3
8	27	1
8	28	2
8	30	1
9	7	48
9	8	20
9	11	10
9	12	3
9	13	19
9	14	44
9	15	31
9	16	6
9	17	40
9	18	49
9	19	37
9	20	20
9	21	41
9	22	14
9	24	13
9	25	9
9	27	3
9	28	7
9	29	19
9	30	2
9	31	8
9	32	4
9	33	2
9	34	4
10	7	7
10	8	14
10	12	17
10	13	3
10	14	4
10	15	20
10	16	30
10	17	18
10	18	36
10	19	83
10	20	78

Max Len	cost	#lib
10	21	14
10	22	30
10	23	26
10	24	35
10	25	4
10	26	17
10	27	3
10	28	21
10	29	9
10	30	19
10	32	2
10	34	5
10	35	1
10	36	2
11	13	4
11	14	1
11	15	1
11	16	45
11	17	54
11	18	20
11	19	47
11	20	42
11	21	89
11	22	6
11	23	24
11	24	20
11	25	21
11	26	18
11	27	39
11	28	22
11	29	20
11	30	11
11	31	5
11	32	3
11	33	3
11	34	1
11	35	1
11	37	1
12	13	1
12	14	5
12	16	2
12	17	4
12	18	11
12	19	3
12	20	12

Table 12: There are $\#lib$ sub-libraries that can synthesize circuits with maximum length = $MaxLen$ and cost = $cost$.

Max Len	cost	#lib
12	21	52
12	22	30
12	23	17
12	24	52
12	25	44
12	26	42
12	27	14
12	28	23
12	29	31
12	30	28
12	31	6
12	32	12
12	33	13
12	34	5
12	35	9
12	36	4
12	37	4
12	38	9
13	16	1
13	18	5
13	19	63
13	20	8
13	21	15
13	22	17
13	23	36
13	24	10
13	25	55
13	26	19

Max Len	cost	#lib
13	27	17
13	28	1
13	29	11
13	30	22
13	31	10
13	32	8
13	33	10
13	34	16
13	35	10
13	36	4
13	37	13
13	38	3
13	39	13
13	40	2
13	41	4
13	42	2
13	47	3
14	16	6
14	18	3
14	19	18
14	20	17
14	21	2
14	22	1
14	23	21
14	24	48
14	25	15
14	26	8
14	27	21

Max Len	cost	#lib
14	28	27
14	29	3
14	30	3
14	31	2
14	32	9
14	34	8
14	35	9
14	36	11
14	37	6
14	38	5
14	39	1
14	40	7
14	41	1
14	42	2
14	45	1
15	21	3
15	23	4
15	24	4
15	25	15
15	26	8
15	28	16
15	29	10
15	30	1
15	31	9
15	32	16
15	33	9
15	34	1
15	35	1

Max Len	cost	#lib
15	36	3
15	38	1
15	39	2
15	41	4
15	42	5
15	43	2
15	44	4
15	45	2
16	23	2
16	25	2
16	26	7
16	30	6
16	31	3
16	32	6
16	33	4
16	34	3
16	36	3
16	37	3
16	38	3
16	41	1
16	46	2
16	47	3
16	48	1
16	49	2
17	25	2
17	35	4
18	33	1
18	36	2

Table 13: Table 12 cont.: There are $\#lib$ sub-libraries that can synthesize circuits with maximum length $= MaxLen$ and cost $= cost$.

Max cost	#lib
0	7
1	36
2	69
3	54
4	138
5	51
6	98
7	102
8	94
9	48
10	31
11	72
12	56
13	18

Max cost	#lib
14	57
15	75
16	45
17	23
18	139
19	153
20	52
21	76
22	92
23	56
24	79
25	180
26	230
27	207

Max cost	#lib
28	227
29	202
30	131
31	139
32	229
33	117
34	95
35	99
36	91
37	104
38	69
39	56
40	37
41	33

Max cost	#lib
42	51
43	25
44	14
45	7
46	5
47	10
48	1
49	3
50	4
51	3
52	1
53	2
54	2

Table 14: There are $\#lib$ libraries that can synthesize circuits with maximum cost $= Maxcost$.

Max cost	Len	#lib
0	1	7
1	1	36
2	2	57
2	4	12
3	3	24
3	4	1
3	5	29
4	4	129
4	6	9
5	1	6
5	6	15
5	7	30
6	2	48
6	6	38
6	8	12
7	3	54
7	7	30
7	8	4
7	9	14
8	4	18
8	5	6
8	6	12
8	8	34
8	9	19
8	10	5
9	5	42
9	10	6
10	3	4
10	4	27
11	3	17
11	4	16
11	5	39
12	4	19
12	5	8
12	6	29
13	5	9
13	6	9
14	3	2
14	4	34
14	6	12
14	7	9
15	3	1
15	4	37
15	5	8
15	6	2

Max cost	Len	#lib
15	7	22
15	8	3
15	9	2
16	4	16
16	5	12
16	6	14
16	7	3
17	5	9
17	6	2
17	7	10
17	8	2
18	4	1
18	5	3
18	6	26
18	7	73
18	8	36
19	5	4
19	6	9
19	7	127
19	8	7
19	9	6
20	5	5
20	6	12
20	7	22
20	8	5
20	9	2
20	10	6
21	6	19
21	7	27
21	8	12
21	10	6
21	11	12
22	6	2
22	7	45
22	8	45
23	6	2
23	7	13
23	8	23
23	9	15
23	10	3
24	6	4
24	7	5
24	8	43
24	9	19
24	10	8

Max cost	Len	#lib
25	6	11
25	7	21
25	8	22
25	9	93
25	10	21
25	11	12
26	6	3
26	7	20
26	8	46
26	9	51
26	10	77
26	11	25
26	12	8
27	7	20
27	8	25
27	9	47
27	10	25
27	11	73
27	12	14
27	13	3
28	8	22
28	9	53
28	10	48
28	11	15
28	12	55
28	13	33
28	14	1
29	7	6
29	8	11
29	9	85
29	10	41
29	11	44
29	12	8
29	14	7
30	8	11
30	9	24
30	10	47
30	11	26
30	12	22
30	15	1
31	7	3
31	8	1
31	9	31
31	10	33
31	11	46

Max cost	Len	#lib
31	12	12
31	13	13
32	8	1
32	9	33
32	10	36
32	11	23
32	12	89
32	13	12
32	14	34
32	15	1
33	9	18
33	10	28
33	11	27
33	12	4
33	13	3
33	14	35
33	15	2
34	9	17
34	10	33
34	11	21
34	12	5
34	13	6
34	14	4
34	15	6
34	16	3
35	9	10
35	10	26
35	11	41
35	12	13
35	13	6
35	16	3
36	9	3
36	10	18
36	11	18
36	12	19
36	13	13
36	14	20
37	10	25
37	11	24
37	12	20
37	13	16
37	14	17
37	15	2
38	10	9
38	11	23

Table 15: There are $\#lib$ libraries that can synthesize circuits with maximum cost = $Maxcost$ and length = Len.

Max cost	Len	#lib
38	12	13
38	13	4
38	14	3
38	15	14
38	16	3
39	10	1
39	11	15
39	12	27
39	13	13
40	10	6
40	11	9
40	12	14
40	13	2

Max cost	Len	#lib
40	14	3
40	18	3
41	11	2
41	12	12
41	13	13
41	14	6
42	11	19
42	12	18
42	13	10
42	14	2
42	15	2
43	11	6
43	12	6

Max cost	Len	#lib
43	13	9
43	14	4
44	11	1
44	13	10
44	14	1
44	15	2
45	13	4
45	14	3
46	12	1
46	14	2
46	15	2
47	12	2
47	13	3

Max cost	Len	#lib
47	14	5
48	13	1
49	14	3
50	13	1
50	15	3
51	14	2
51	15	1
52	15	1
53	15	2
54	15	2

Table 16: Table 15 cont.: There are $\#lib$ libraries that can synthesize circuits with maximum cost = $Maxcost$ and length = Len.

two examples from the 1960 universal reversible sub-libraries. Table 15 continued in Table 16 shows the same results as Table 14 with more details on the cost of the circuits synthesized by these sub-libraries.

The above results show that there are 1960 sub-libraries, each can be used as a universal reversible gate library:

1. The sub-library that synthesize the best maximum length circuits is the main library,

$$\{N_1, N_2, N_3, F_{12}, F_{13}, F_{23}, F_{21}, F_{32}, F_{31}, T_{123}, T_{132}, T_{321}\},$$

   where the best maximum length = 8 gates with cost = 20 and maximum cost = 23 with circuit length =7.

2. The sub-library that synthesize the best maximum cost circuit is,

$$\{N_1, N_2, N_3, F_{12}, F_{13}, F_{23}, F_{21}, F_{32}, F_{31}, T_{132}, T_{321}\},$$

   where the best maximum cost = 22 with circuit length =7 and the maximum length = 9 gates with cost = 11.

3. The sub-library that synthesize the worst maximum length circuits is,

$$\{N_3, F_{32}, F_{31}, T_{123}\},$$

   where the worst maximum length = 18 gates with cost = 36 and maximum cost = 40 with circuit length =18.

4. The sub-library that synthesize the worst maximum cost circuit is,

$$\{N_1, F_{12}, T_{123}, T_{321}\},$$

   where the worst maximum cost = 54 with circuit length =15 and the maximum length = 16 gates with cost = 49.

Non of the 1960 universal reversible gate libraries can synthesize a circuit with the best maximum cost which is 17 as shown in Table 8. This best maximum cost comes from sub-libraries that are not universal, for example, the sub-library,

$$\{N_1, N_2, F_{13}, F_{23}, F_{31}, T_{123}, T_{321}\},$$

is not universal since it can syntheise circuits for 1151 specifications only, where its best maximum cost = 17 with circuit length =7 and the maximum length = 7 gates with cost = 8.

5 Conclusion

By reducing the representation of the reversible circuit synthesis problem to permutation group, Schreier-Sims Algorithm for the strong generating set-finding problem is used to put tight bounds on the synthesis of 3-bit reversible circuits using the NFT library. Using group-theory algebraic software GAP shows that,

1. The minimum length of a circuit ranges from 1 to 8 gate(s) with average length = 5.865.
2. The maximum length of a circuit ranges from 7 to 18 gates with average length = 14.639.
3. The minimum cost of a reversible circuit ranges from 1 to 17 with average cost = 11.769.
4. The maximum cost of a reversible circuit ranges from 18 to 54 with average cost = 40.759.

The analysis shows that there 1960 universal reversible sub-libraries from the NFT library. The upper and lower bounds on the length of the circuits come from using the universal reversible sub-libraries while the upper bound on the cost of the circuits comes from using the universal reversible sub-libraries, while the lower bounds on the cost of the circuits comes from other sub-libraries which are not necessary universal.

The same sort of analysis is applied to other libraries such as NFP, NFFr, NFPT, NFTFr and NFPFr. The tight bounds for these libraries are under preparation.

References

[1] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Phys. Rev. A*, 52(5):3457–3467, Nov 1995.

[2] C. Bennett. Logical reversibility of computation. *IBM Journal of Research and Development*, 17(6):525–532, 1973.

[3] T. Beth and M. Roetteler. Quantum algorithms: Applicable algebra and quantum physics. In *Quantum Information*, pages 96–150. Springer, 2001.

[4] J.D. Dixon and B. Mortimer Permutation groups. New York: Springer, 1996.

[5] A. De Vos, B. Desoete, A. Adamski, P. Pietrzak, M. Sibinski, and T. Widerski. Design of reversible logic circuits by means of control gates. In *Proceedings of the 10th International Workshop on Integrated Circuit Design, Power and Timing Modeling, Optimization and Simulation*, pages 255-264, 2000.

[6] A. De Vos, B. Raa and L. Storme Generating the group of reversible logic gates. Journal of Physics A: Mathematical and General, 35(33): 7063–7078, 2002.

[7] A. De Vos and Y. V. Rentergem From Group Theory to Reversible Computers. International Journal of Unconventional Computing, 4(1): 79-88, 2008.

[8] A. De Vos and S. De Baerdemacker Symmetry Groups for the Decomposition of Reversible Computers, Quantum Computers, and Computers in between. Symmetry, 3(2): 305-324, 2011.

[9] A. De Vos, B. Desoete, F. Janiak, and A. Nogawski. Control gates as building blocks for reversible computers. In *Proceedings of the 11th International Workshop on Power and Timing Modeling, Optimization and Simulation*, pages 9201–9210, 2001.

[10] G. W. Dueck and D. Maslov. Reversible function synthesis with minimum garbage outputs. In *Proceedings of the 6th International Symposium on Representations and Methodology of Future Computing Technologies*, pages 154–161, 2003.

[11] E. Fredkin and T. Toffoli. Conservative logic. *International Journal of Theoretical Physics*, 21:219–253, 1982.

[12] The GAP Group. GAP – Groups, Algorithms, and Programming, Version 4.6.3; 2013. (http://www.gap-system.org)

[13] O. Golubitsky and S.M. Falconer and D. Maslov. Synthesis of the optimal 4-bit reversible circuits. In *Proceedings of the 47th Design Automation Conference*, pages 653-656, 2010.

[14] J. Gruska. *Quantum Computing*. McGraw-Hill, London, 1999.

[15] K. Iwama, Y. Kambayashi, and S. Yamashita. Transformation rules for designing CNOT–based quantum circuits. In *Proceedings of the 39th Conference on Design Automation*, pages 419–424. ACM Press, 2002.

[16] M.I. Kargapolov and Ju.I. Merzljakov Fundamentals of the theory of groups. Berlin: Springer, 1979.

[17] D. Maslov, G. W. Dueck, and D. M. Miller. Fredkin/Toffoli templates for reversible logic synthesis. In *Proceedings of the ACM/IEEE International Conference on Computer-Aided Design*, page 256, 2003.

[18] D. Maslov, G. W. Dueck, and D. M. Miller. Simplification of Toffoli networks via templates. In *Proceedings of the 16th Symposium on Integrated Circuits and Systems Design*, page 53, 2003.

[19] D. Maslov and C. Young and D. M. Miller and G. W. Dueck. Quantum circuit simplification using templates. Design, Automation and Test in Europe, pages 1208-1213, 2005.

[20] D. Maslov. Reversible logic synthesis benchmarks. [Online]. Available: http://www.cs.uvic.ca/∼dmaslov/.

[21] D. M. Miller and G. W. Dueck. Spectral techniques for reversible logic synthesis. In *Proceedings of the 6th International Symposium on Representations and Methodology of Future Computing Technologies*, pages 56–62, 2003.

[22] D. M. Miller, D. Maslov, and G. W. Dueck. A transformation based algorithm for reversible logic synthesis. In *Proceedings of the 40^{t}h Conference on Design Automation*, pages 318–323, 2003.

[23] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, United Kingdom, 2000.

[24] K. N. Patel, I. L. Markov, and J. P. Hayes. Efficient synthesis of linear reversible circuits. *arXiv e-Print quant-ph/0302002*, 2003.

[25] A. K. Prasad and V. V. Shende and K. N. Patel and I. L. Markov and J. P. Hayes. Data structures and algorithms for simplifying reversible circuits. J. Emerg. Technol. Comput. Syst., 2(4), October 2006.

[26] D. Simon. On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 116–123, 1994.

[27] A. Seress Permutation Group Algorithms. Cambridge University Press, 2002.

[28] L. Storme, A. De Vos, G. Jacobs Group Theoretical Aspects of Reversible Logic Gates. Journal of Universal Computer Science, 5(5): 307–321, 1999.

[29] T. Toffoli. Reversible computing. In W. de Bakker and J. van Leeuwen, editors, *Automata, Languages and Programming*, page 632. Springer, New York, 1980. Technical Memo MIT/LCS/TM-151, MIT Lab for Computer Science (unpublished).

[30] G. Yang, X. Song, W. N.N. Hung, M. A. Perkowski, and C.-J. Seo, Synthesis of reversible circuits with minimal costs. *CALCOLO*, 45:193–206, 2008.

[31] A. Younes and J. Miller. Representation of Boolean quantum circuits as Reed-Muller expansions. *International Journal of Electronics*, 91(7):431–444, 2004.